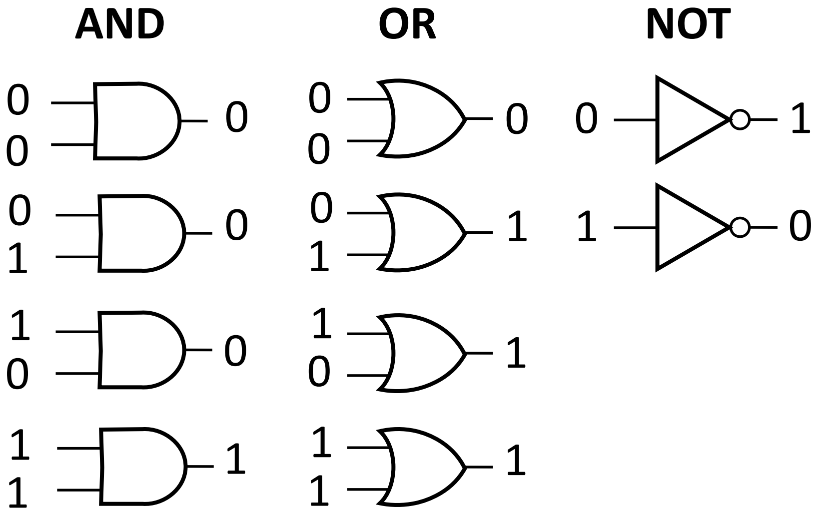
EECS 270 Fall 2022
Introduction to Logic Design

2. Bits Everywhere!

Recap 1/3

- Digital circuit inputs and outputs have only two ``valid'' states:
  - Low/Off/False/0
  - High/On/True/1

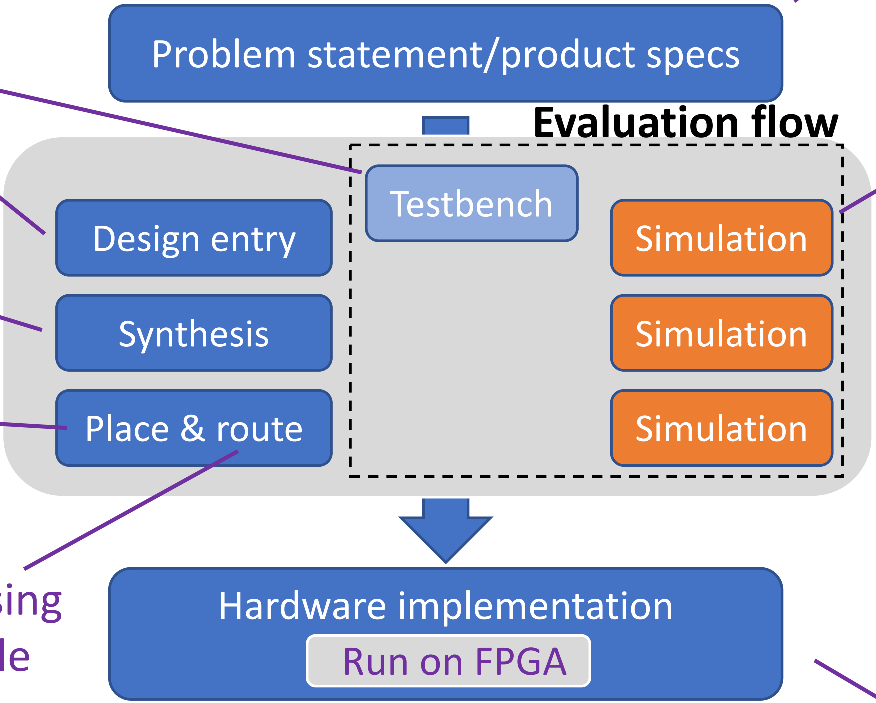
- Basic Boolean logic operators:

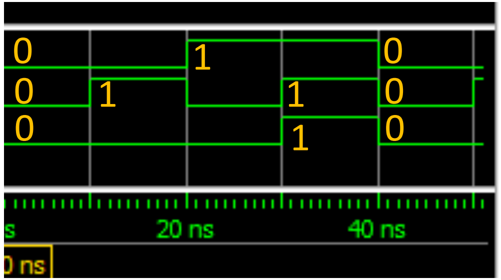


Recap 2/3

provided

Verilog





Turns design entry into a netlist

Places synthesis nets into available locations on the target device

Connects physical nets using routing resources available on the target device

requested

Recap 3/3

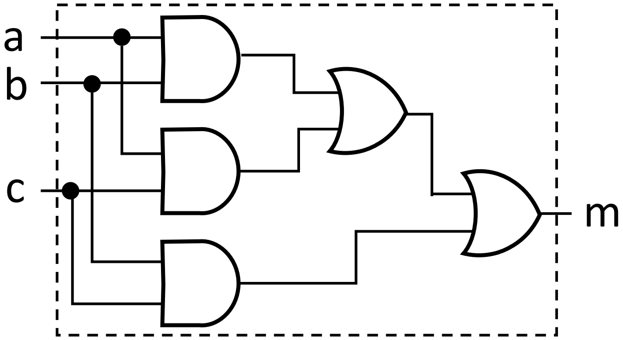
- Combinational logic design process

```
// Behavioral description of majority over 3 Boolean inputs

module MAJ (a, b, c, m);
    input a, b, c;
    output m;

    assign m = a && b || a && c || b && c;
endmodule
```

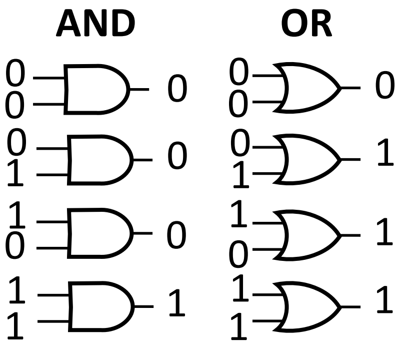
- Truth table:

a	b	c	m
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

- Schematic of MAJ:



- Boolean operators:



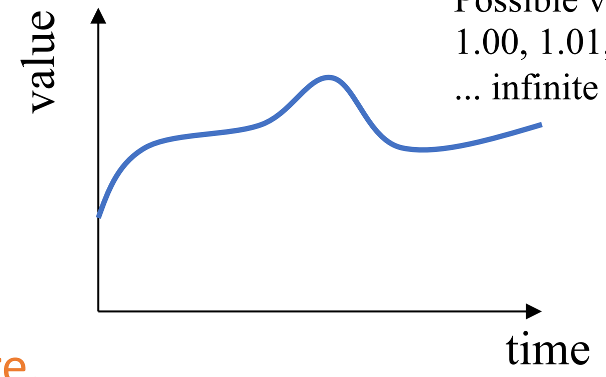
Today's lecture

- Digital circuit inputs and outputs have only two ``valid'' states:
  - Low/Off/False/0
  - High/On/True/1

- Why digital?
- Why binary?
- How do we represent arbitrary data with only 0s and 1s?
- How are Boolean operators implemented in real hardware?

Analog vs Digital

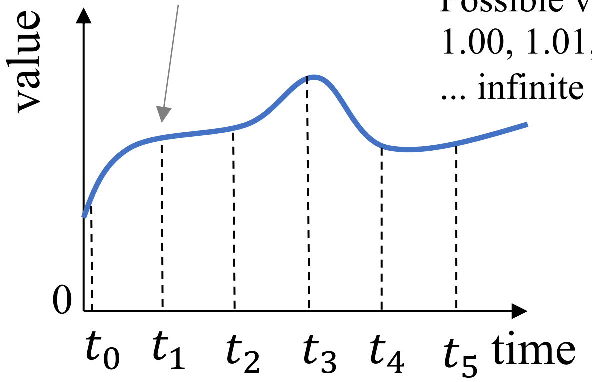
- **Analog signals:**
  - An analog signal is continuous in both time and amplitude.
  - Analog signals in the real world include current, voltage, temperature, pressure, light intensity, and so on.



Possible values:
1.00, 1.01, 2.0000009,
... infinite possibilities

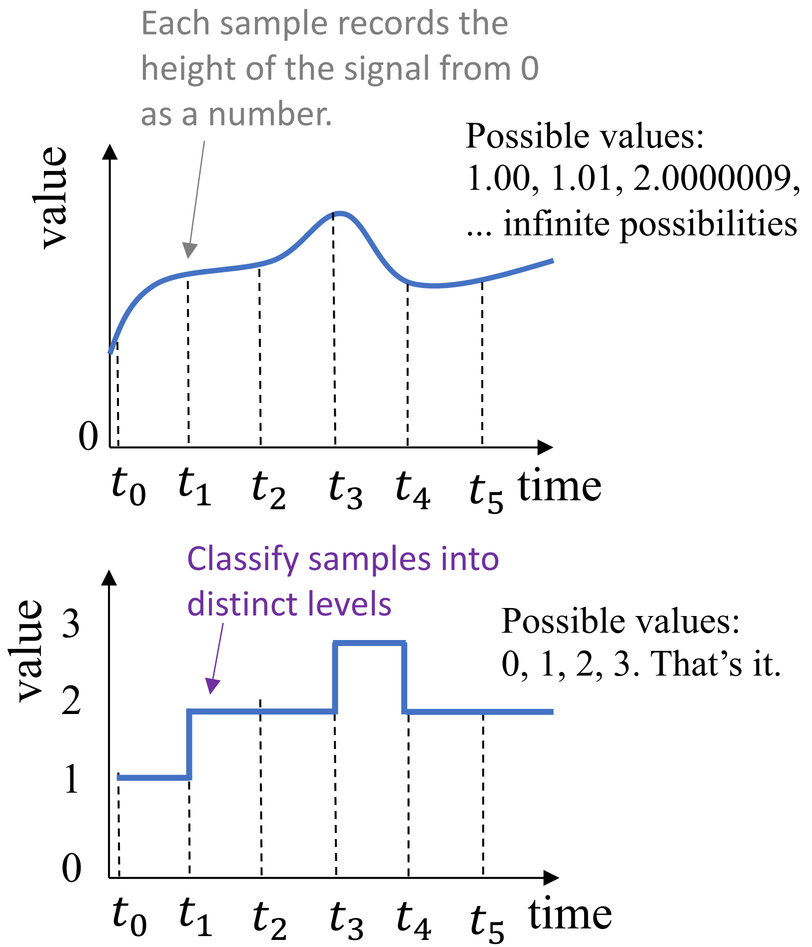
Analog vs Digital

- **Digital signals:**
  - The digital signal contains the digital values converted from the analog signal at the specified time instants.

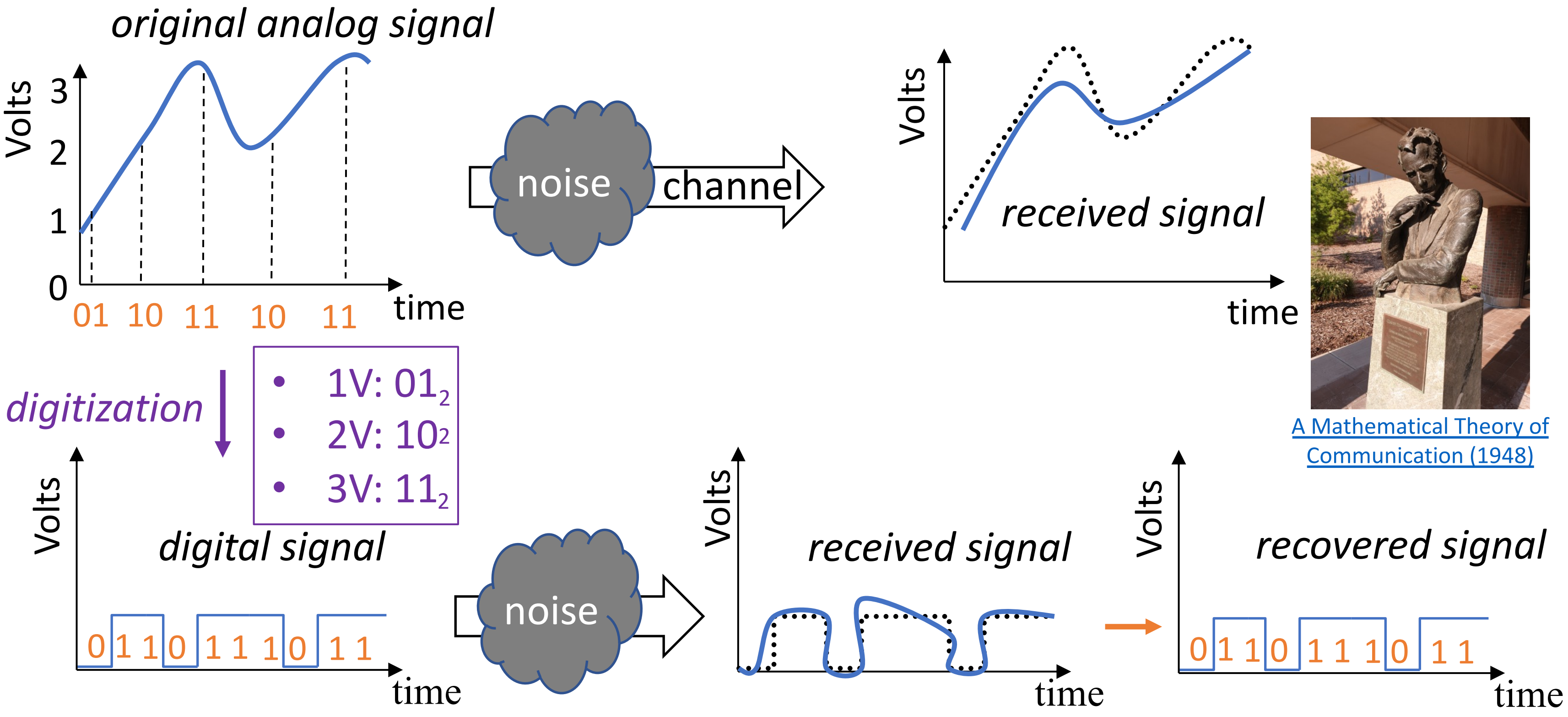


Analog vs Digital

- **Digital signals:**
  - The digital signal contains the digital values converted from the analog signal at the specified time instants.
  - Digital signals can take a finite number of values.



Digitization benefits: noise tolerance



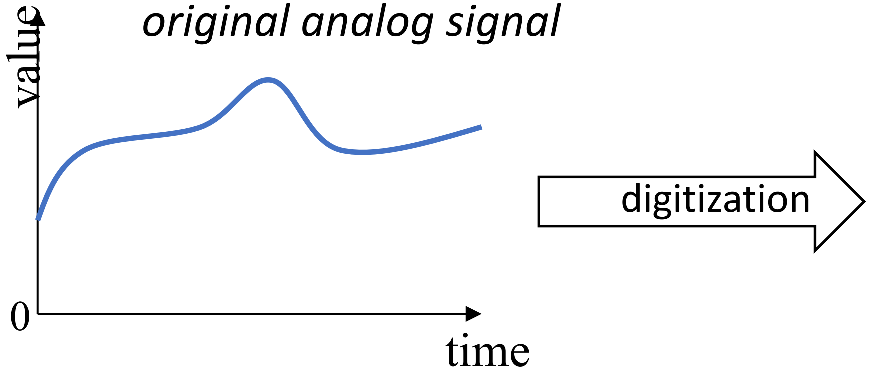
[A Mathematical Theory of Communication (1948)]

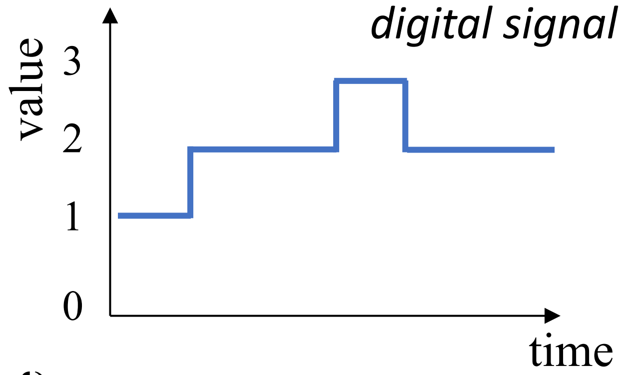
Digitization benefits: compression, etc.

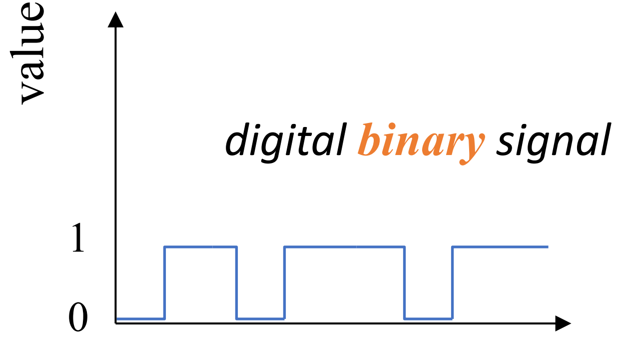
8x10 = 80 bits		**44 bits**
00000000		00
00000000		00
00001100		1000001100
00010000		1000010000
00000000	compression →	00
00000000		00
00000000		00
11111111		01
11111111		01
10000101		1010000101

binary digi**ts**

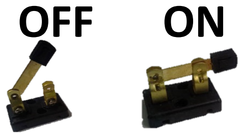
Why binary?







- Computer are made of billions of tiny switches



- Storing/transmitting one of *two* values is easier than three or more (e.g., loud beep or quiet beep, light or no light)

Data encoding

- ❑ Each position represents a quantity
- ❑ The symbol in each position indicates how many of that quantity

$$\overset{2\,1\,0}{249}_{10} = 2 * 10^2 + 4 * 10^1 + 9 * 10^0$$

base

64 8

$$\overset{2\,1\,0}{371}_8 = 3 * 8^2 + 7 * 8^1 + 1 * 8^0 = 249_{10}$$

192 + 56 + 1

$$\overset{7\,6\,5\,4\,3\,2\,1\,0}{11111001}_2 = ?_{10}$$ (Question 1)

=

Data encoding

- Each position represents a quantity
- The symbol in each position indicates how many of that quantity

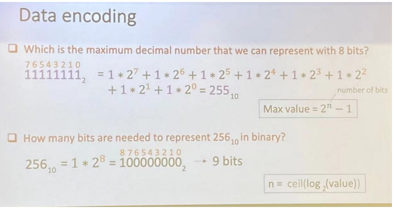
$$\overset{2\,1\,0}{249}_{10} = 2 * 10^2 + 4 * 10^1 + 9 * 10^0$$

base

$$\overset{2\,1\,0}{371}_8 = 3 * 8^2 + 7 * 8^1 + 1 * 8^0 = 249_{10}$$

$$\overset{7\,6\,5\,4\,3\,2\,1\,0}{11111001}_2 = ?_{10}$$ (Question 1)

$$1 * 2^7 + 1 * 2^6 + 1 * 2^5 + 1 * 2^4 + 1 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0 = 249_{10}$$


Data encoding

- Which is the maximum decimal number that we can represent with 8 bits?

$$\overset{7\,6\,5\,4\,3\,2\,1\,0}{11111111}_2 = 1 * 2^7 + 1 * 2^6 + 1 * 2^5 + 1 * 2^4 + 1 * 2^3 + 1 * 2^2 + 1 * 2^1 + 1 * 2^0 = 255_{10}$$

number of bits

$\boxed{\text{Max value} = 2^n - 1}$

- How many bits are needed to represent 256_{10} in binary?

$$256_{10} = 1 * 2^8 = \overset{8\,7\,6\,5\,4\,3\,2\,1\,0}{100000000}_2 \rightarrow 9 \text{ bits}$$

$\boxed{n = \text{ceil}(\log_2(\text{value}))}$

Question 2: Which is the maximum decimal number that we can represent with 8 bits?

Question 3: How many bits are needed to represent 256_{10} in binary?

UNIVERSITY OF MICHIGAN

Data encoding

- ❑ Text
  - ➢ ASCII: 7- (or 8-) bit encoding of each letter, number, or symbol

Encoding	Symbol
010 0000	<space>SP
010 0001	!
010 0010	"
010 0011	#
010 0100	$
010 0101	%
010 0110	&
010 0111	'
010 1000	(
010 1001	)
010 1010	*
010 1011	+
010 1100	,
010 1101	-
010 1110	.
010 1111	/

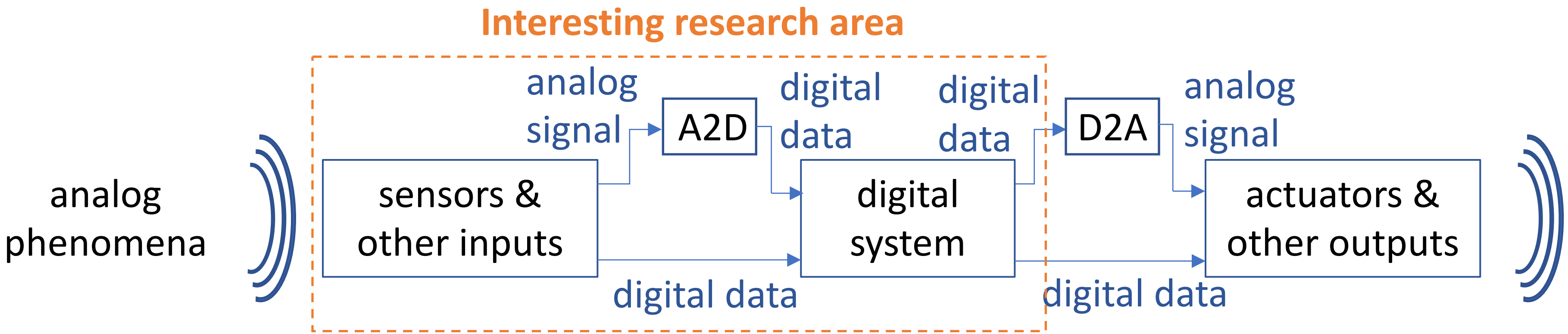
Encoding	Symbol
100 0001	A
100 0010	B
100 0011	C
100 0100	D
100 0101	E
100 0110	F
100 0111	G
100 1000	H
100 1001	I
100 1010	J
100 1011	K
100 1100	L
100 1101	M

Encoding	Symbol
100 1110	N
100 1111	O
101 0000	P
101 0001	Q
101 0010	R
101 0011	S
101 0100	T
101 0101	U
101 0110	V
101 0111	W
101 1000	X
101 1001	Y
101 1010	Z

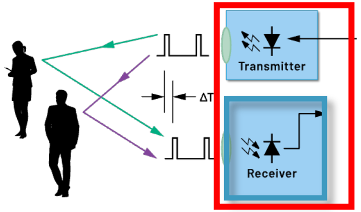
Encoding	Symbol
110 0001	a
110 0010	b
...	
111 1001	y
111 1010	z
011 0000	0
011 0001	1
011 0010	2
011 0011	3
011 0100	4
011 0101	5
011 0110	6
011 0111	7
011 1000	8
011 1001	9

$$\underbrace{1010010}_{R}\underbrace{1000101}_{E}\underbrace{1010011}_{S}\underbrace{1010100}_{T}$$

Real world setup

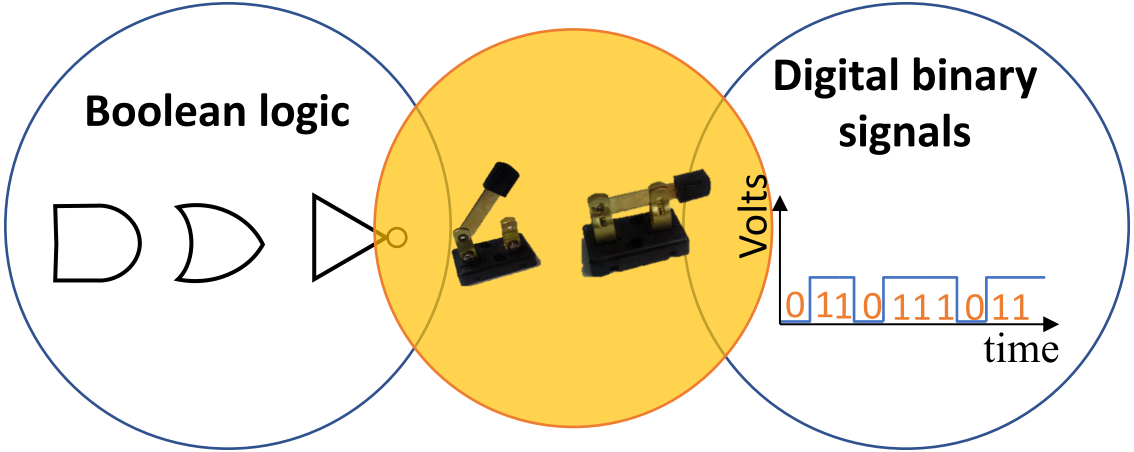


Time-of-flight sensor



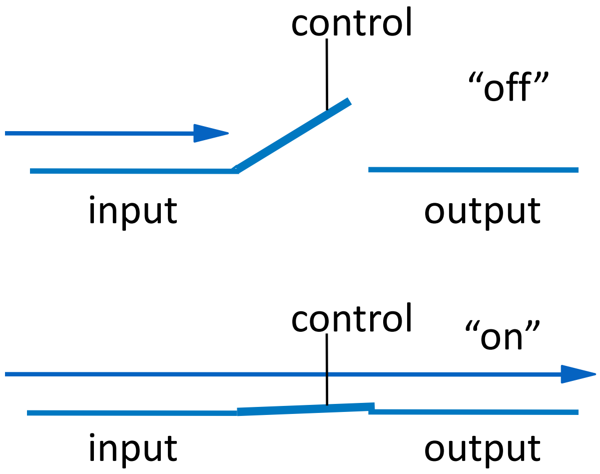
measures round trip time

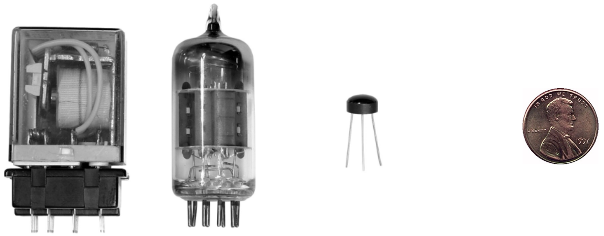
Bird’s-eye view



Switches

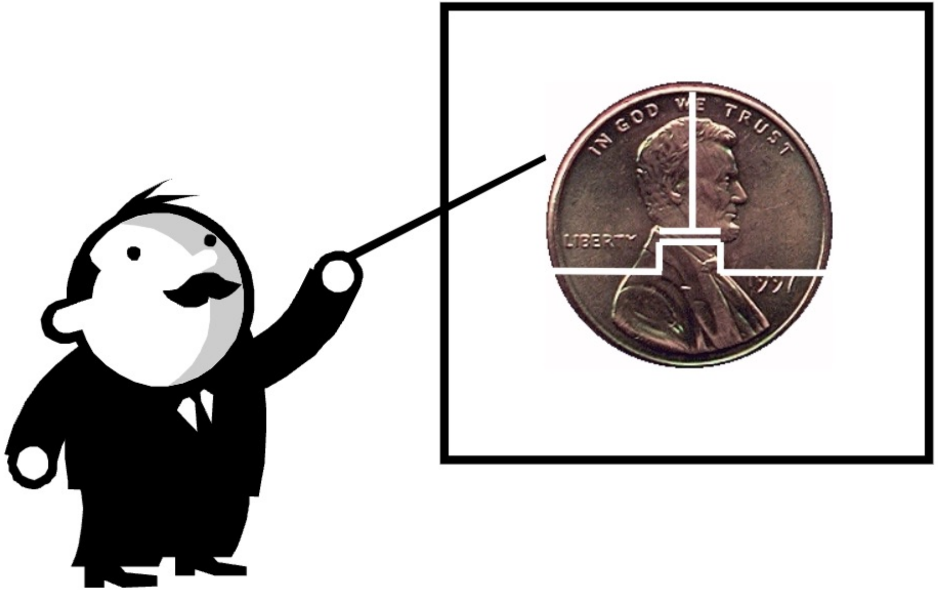
- A switch has three parts:
  - Input
  - Output
  - Control





Relay Tube Transistor Penny

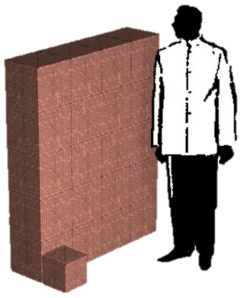
Transistor scaling



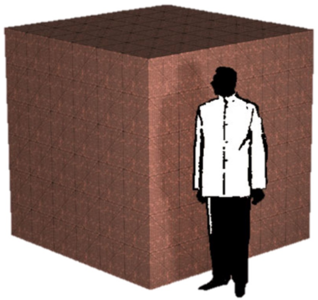
Courtesy of T. Sherwood

Transistor scaling

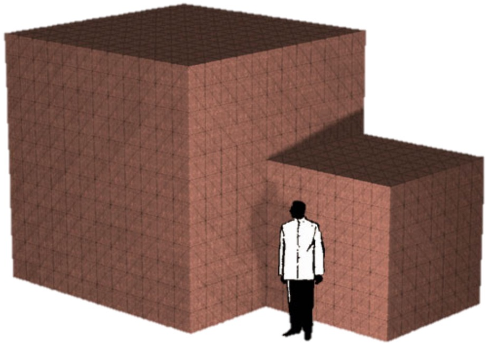
- One million transistors



- Ten million transistors



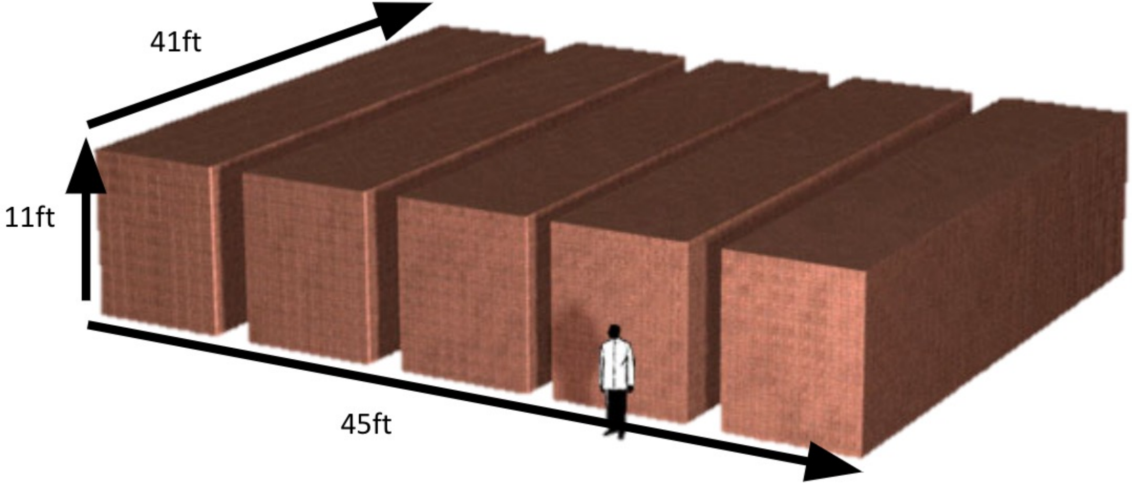
- One hundred million transistors



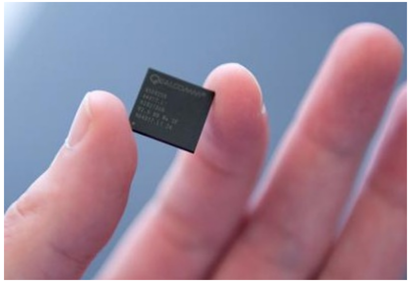
Courtesy of T. Sherwood

Transistor scaling

- One billion transistors



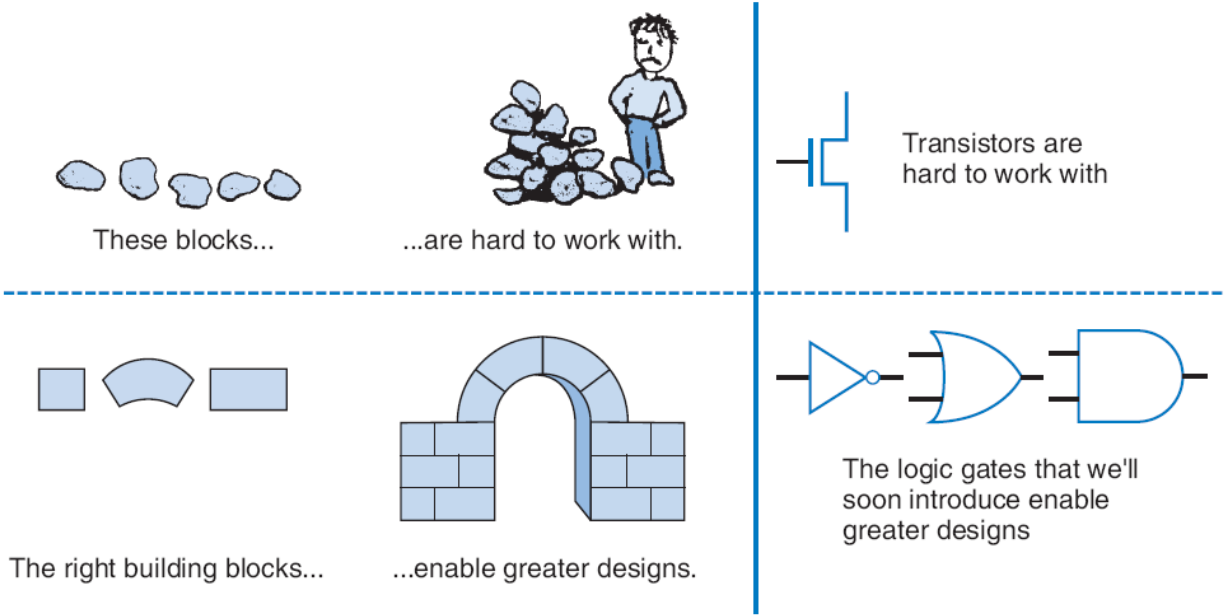
Four billion transistors



Courtesy of T. Sherwood

From transtors to gates

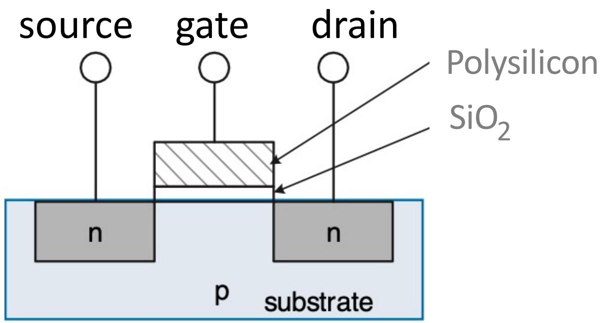
- Switches are hard to work with …

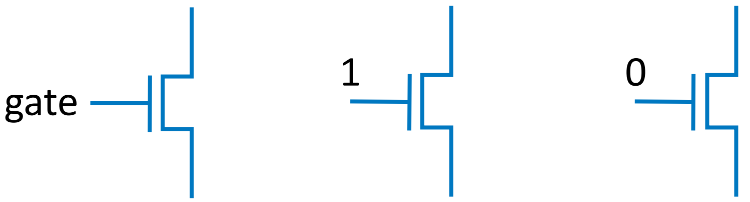


Transistors basics

- Gate serves as controller
- Current flows from source to drain or vice versa

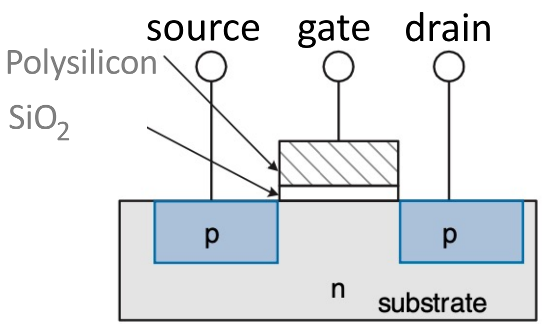
❑ n-channel metal-oxide semiconductor (nMOS)

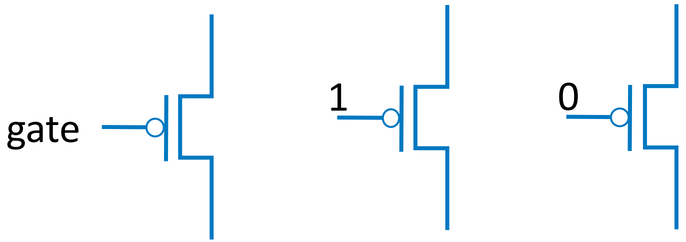




nMOS symbol | nMOS conducts | nMOS does not conduct

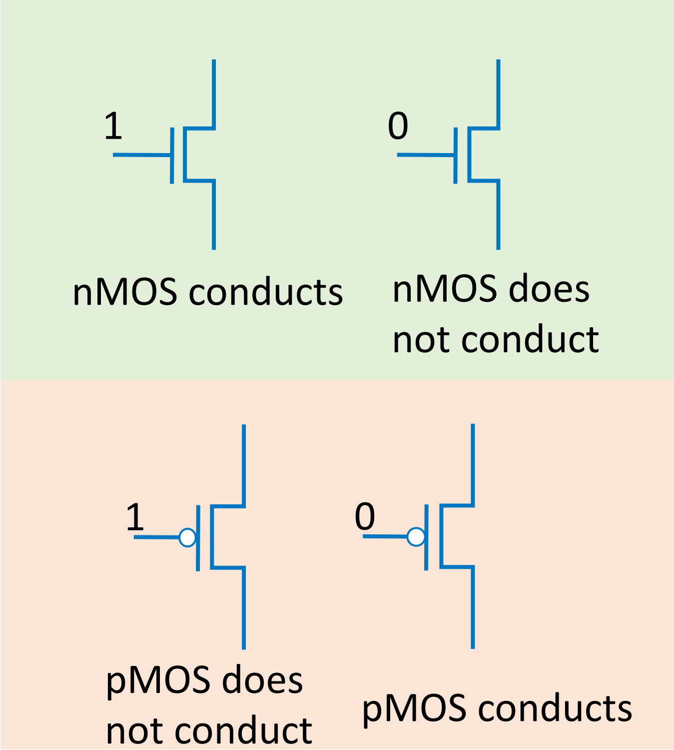
❑ p-channel metal-oxide semiconductor (pMOS)

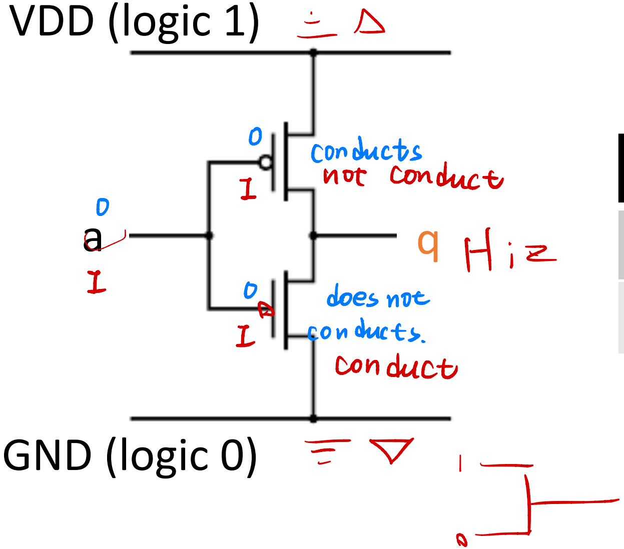




pMOS symbol | pMOS does not conduct | pMOS conducts

Last exercise of the day





Fill out the truth table

a	q
0	? 1
1	?

Questions?